EE175AB Final Report Template

Smart Battery Charger

EE 175AB Final Report

Department of Electrical Engineering, UC Riverside

Project Team Member(s)	Gilberto Peraza-Martinez, Rohan Badiga, Min Hua Wu
Date Submitted	03/15/2021
Section Professor	Roman Chomko
Revision	1.0
URL of Project Wiki/Webpage	https://github.com/Agentrb241/Portfolio/tree/main/Final_Design_Project_Battery_Charger https://youtu.be/IO5YxovEinE
Permanent Emails of all team members	rohanbadiga@gmail.com john@omcl.com.tw perazamgilberto@gmail.com

Summary

This report presents a Battery charger that has successfully charged a Nickel Cadmium 1.2V Battery

Note:

- **Sections marked with * are required**
- **In each section, you must clearly identify which team member is responsible for which objectives, modules or tasks.**

Revisions

Version	Description of Version	Author(s)	Date Completed	Approval
Version Number	Information about the revision. This table does not need to be filled in whenever a document is touched, only when the version is being upgraded.	names	00/00/00	
0.5	First draft	Rohan Badiga, Min-Hua Wu, Gilberto Peraza-Martinez	03/15/2021	

This template serves as a basis for the EE175 Senior Design Project Final Report.

This document should be customized to the needs of each specific project. This template is only a starting point.

This template does not imply that all components listed in this template should be included (except the sections marked with *) nor does it imply that this template includes all the necessary components needed for a specific project. Instead this template provides a basic starting point that will work well in many situations. The design team must modify and/or expand the contents in order to meet the specific requirements of their project.

REMOVE the sections (without *) that do not apply to your project and renumber the sections

You can change the section titles to better match your project.

Table of Contents

1 * Executive Summary

Battery charging is seen as a necessity in today's society, especially with today's technology moving towards a greener footprint and electric modes of transportation on the rise. Our team has challenged themselves to create a battery charger that is not only safe but cost effective as well. Our goal was to design a prototype that is able to charge a 1.2 V 1000mAh Nickel Cadmium battery in approximately 3 to 4 hours. The reason for this is that the general public views batteries as single use and throws them away after a single use despite them being able to be reused many times. We endeavored to make a smart battery charger that is safe which is shown by the 8 relies we used to ensure that if the system began to fail we would be able to mitigate the damage done to both the product and the consumer. In addition to charging the battery, our prototype has a smart user interface. Utilizing the Arduino we are able to utilize sensors and identify temperatures at critical points and see sensor outputs to make sure the charging profile of the battery is being maintained. Through the use of temperature sensors if any part of the project began to overheat the relays would disconnect the power sources from the circuit, a valuable safety feature. Moreover, using the Arduino's built in Serial Monitor we can control the product and see all values read by the sensors. We were able to create a battery charger that charged a battery at ~1/10C and this was important as this shows if given more time we would have been able to to produce a product with a faster charge rate. The constant voltage circuit is also capable of charging batteries that are not NiCd batteries due to their specific kind of charging profile. The voltage received at the load is ~1.53 with a variance of .65% (.01V) and because of that this circuit is also a valuable, functioning part of our project.

2 * Introduction

2.1 * Design Objectives and System Overview

Gilberto Peraza-Martinez:

Designing a battery charger was an appealing project because it did not necessarily play to the strengths of the group and was a branch of electrical engineering that was not covered in depth in school: power engineering. We did not participate in particularly high voltages/currents throughout our college careers but this project we learned to become very cautious and wary of how power is generated and dissipated in circuits. The two main circuits that were instrumental were the constant current circuit (CCC) and the constant voltage circuit (CVC) which were difficult circuits to perfect and in the construction of these circuits we learned that testing it with lower voltages that intended resulted in safer experimentation. This project taught us a lot, especially on how to be safe when applying untested knowledge. It also gave me greater insight into how the stable currents and voltages at the loads can be maintained even without the specific circuits that we implemented here.

The constant current circuit was built out of darlington transistor pairs rather than current mirrors which is what we were going to build at first. Upon investigating the feasibility of using current mirrors we realized that temperature can vary greatly with the higher power dissipation that the second transistor (Q2). Temperature can greatly affect the matching of Q2 to the first transistor (Q1) and if temperature in the second transistor begins to rise then something called thermal runaway may occur resulting in a further increase in both temperature and power dissipation in Q2, putting our battery at risk.

Choosing darlington pairs make sense because there is no expected need for switching at high frequencies for our specific purposes. There is also an increase in collector current by a factor of Beta and the next increase in collector of the following transistor is also amplified by that transistor's beta value leading to a much higher emitter current for the second transistor. There is not a similar concern of thermal runaway because the transistors do not have to be perfectly matched in order for the current to continue working.

The entirety of the design of the constant current circuit was done by me and getting the constant current circuit to work was my responsibility. Using darlington pairs was an efficient way of achieving high current gain and having stable current at the load. We were only able to achieve currents of ~1/10 C (100 mA) which is not in the area of fast charging that we were hoping to achieve. The sensors that we use are not very accurate as they are inexpensive voltage sensors that are used in conjunction with calculations done in the code to find the current entering the load. These sensors lack precision and as such the error of our sensor data is within 10%.

My design of the voltage circuit (a buck converter)was implemented by John and the reason we chose the buck converter was because, to our existing knowledge it is the cheapest and best way to provide a constant voltage to the load. The buck converter was an incredibly difficult circuit to build and running it at the intended 12 Volts led to multiple test failures. We learned that the reason for our failure was the lack of components rated for the correct wattage. In order to overcome that I instead used multiple resistors of lower values to add up to the correct values that were expected. The design of the VCC resulted in a circuit that delivers 1.53V to the load, with a variance of .01V (~.65%).

The system was designed with electrical isolation in mind so there are 8relays used to ensure that the battery would be easy to isolate in case the system were to fail in some way. We monitored the system using voltage sensors, temperature sensors, and current sensors(we used voltage sensors for this as well). If the temperature of the battery of any elements of the circuit were to exceed 35C the entire system would shut down.

Min Hua Wu:

During the design of this battery charger project, we learned a lot about different parts of electrical engineering that was not taught at school. Before the start of this project, none of us had any idea how a rechargeable battery works and the science behind it; however, after this project, we were able to safely charge a nickel-cadmium battery. Not only did we learn more about batteries, we also learned about power electronics and more specifically use of Buck converter and other important circuitry concepts like the Darlington Pair transistors. The rechargeable battery that we got supplied 1000mA/h, and through constant current charging, our team was able to achieve the goal of charging rate of C/10 which is around 100mA using the constant current circuit.

For my part of the project, I implemented the voltage circuit using the buck converter. The reason why we chose it was because that buck converter was the easiest way to step down voltage in order to provide stable voltage charging; however, the actual implementation took us some time as we could not run the circuit at the desired input voltage (12V). The microcontroller unit is in charge of the switching between constant current charging and constant voltage charging.

Our original goal was to design a battery charger that can support up to charging at least two batteries at a time; however, given the limited time constraints and Covid-19 restrictions, we were not able to achieve that goal.

Rohan Badiga:

Overall the design of the Microcontroller was very critical for the design of the project. It ensured all parts are working to design specifications. Designing the different test cases and the different peripheral implementations made sure our project was successful. My part of the project was to handle the Microcontroller. I made all the code and made the product design for all peripherals.

2.2 * Backgrounds and Prior Art

There are a multitude of battery chargers available for NiMHor NiCd batteries. These other designs are able to charge multiple batteries at once and can deliver higher currents at the load. The description of these battery chargers do not explicitly say that they are smart battery chargers but given the fact that they come from large battery companies it can be assumed that they are.

2.3 * Development Environment and Tools

We did all our testing within the UCR EE labs and we used an oscilloscope to measure output, power supply to power the circuit, and function generator to simulate square wave input. The microcontroller that we used is an Arduino Uno connected with two different charging circuits (one for constant current charging and one for constant voltage charging) and a discharge circuit. The circuits are built with microelectronic components such as resistors, transistors, capacitors, inductors, and amplifiers. Prior to actually building them in person we designed them on LTspice and were able to verify if they would work before using them in the real world.

2.4 * Related Documents and Supporting Materials

1. https://batteryuniversity.com/learn/article/charging_nickel_based_batteries

2.5 * Definitions and Acronyms

Constant Current Circuit (CCC), Constant Voltage Circuit (CVC), Microcontroller Unit (MCU)

3 * Design Considerations

In order to safely charge the battery, we must choose a C rate that is lower than 1C.

3.1 * Realistic Constraints

Since the rechargeable nickel-cadmium battery that we chose supplied 1000mA/h, we were able to achieve a charging rate of C/10 which is 100mA/h and it would take around 10 hours to fully charge, without any loss being taken into account. The current that the CCC gives is around 100mA/h, which is safe and within our specification. The voltage that the CVC gives is around 1.5V; however, the voltage circuit is designed to charge different battery types such as lithium-ion batteries. When charging the nickel-cadmium battery we would only be using the CCC because the charging profile only allows for constant current.

This last year was very hard for our entire group for a multitude of reasons. The Covid-19 placed constraints on us all and we struggled to breach the gap both physically and faced multiple time constraints. We were meant to limit time spent together and finding an appropriate place for us to meet was difficult due to most of the group not residing in Riverside. Buying parts was affected as well due to political reasons and that affected how quickly we received parts necessary for our project to move forward in a timely manner.

3.2 System Environment and External Interfaces

Currently our battery charger supports nickel-cadmium batteries and the MCU must be connected to a computer in order to access the serial monitor interface within Arduino IDE. Within the serial monitor, one can follow the prompt provided to not only choose which circuit to turn on, but also important information such as temperatures, voltages, and currents.

3.3 * Industry Standards

We used LTSpice for all the simulations shown in the report.

3.4 * Knowledge and Skills

Min Hua Wu: Electronic Circuits (EE100AB), Logic Design (EECS120A), Introduction to Embedded Systems (EECS120B)

New Concepts: Charging profile of Nickel Cadmium batteries, Charging rate of batteries (C-rate), Darlington Pair Transistors, Buck Converter, Arduino Coding

Rohan Badiga: EE1A, EE1B, Electronic Circuits (EE100AB), Logic Design (EECS120A), Introduction to Embedded Systems (EECS 120B), Intermediate Embedded Systems (EECS 122A)

New Concepts: Charging profile of Nickel Cadmium Batteries, Darlington Pair Transistors, How to Code and Arduino, How to use various sensors (temperature, voltage, voltage for current, relay)

Gilberto Peraza-Martinz: Electronic Circuits (EE100A & B), EE1A & 1BIntroduction to Embedded Systems (EECS120B), Charging profile of Nickel Cadmium batteries, Buck Converter,

I learned a lot about batteries and the fact that each has a particular charging profile that is important to its continued use and health. Making decisions on what kind of circuits to use made me learn a lot about Darlington Pair Transistors and their advantages and disadvantages. Dealin with such sensitive loads I

learned how to be safe and use low voltages for real life applications and to verify as much as possible with simulations.

3.5 * Budget and Cost Analysis

https://docs.google.com/spreadsheets/d/1_V3RKCp5uimWj2p9UfXKnk9x5e3s4Vab_njzasZhubg/edit#gid=0

3.6 * Safety

Since safety is our number one priority, we did our tests in the UCR lab while strictly following the UCR EE lab safety rules and also Covid-19 restrictions. We wore closed toed shoes at all times and masks while keeping the safe social distance. Because our product is a battery charger, our goal is to make sure that the battery charger does not have any thermal issues or potentially cause a fire. To address this issue, we attached some temperature sensors within our circuits near some of the amplifiers and battery to make sure it does not overheat. In the development of the charger we also used lower voltages until we were ready to accept the risk that a higher voltage posed to the circuit.

3.7 Performance, Security, Quality, Reliability, Aesthetics etc.

Due to the limited time that we have to complete this project, we could only achieve charging one nickel-cadmium battery at a time; furthermore, more safety testings can be done with a longer duration of charging time. Because this is only a prototype, the circuit is completed on a breadboard instead of a printed circuit board (PCB), and components are prone to more external variables such as moving the circuits, some diodes or resistor legs touching...etc. However, we were able to achieve constant current charging and constant voltage charging while closely matching the simulation results.

3.8 * Documentation

Rohan Badiga:
https://drive.google.com/file/d/1lf5H8cfa16ShKucsx7ixLV_Dnqi3hGKO/view?usp=sharing

Min Hua:

https://drive.google.com/drive/folders/1ib1d_0ql3C5ugIxvqt72KIqHUi9RiEKZ?usp=sharing

3.9 Risks and Volatile Areas

With the considerable risk that batteries pose in terms of overheating and the dangerous contents they contain we had to be very careful. We implemented each part of the circuit only after the simulation suggested that things would go well but as we know simulations do not always match real life. We used resistors as loads instead of batteries and only after being sure that the circuit was functioning properly did we use the battery in the circuit.Some volatile areas of our project would be the resistors overheating from excessive power dissipation. We invested in resistors rated for 3W and those made a huge difference in terms of heat dissipation for the resistors at the load in both the CCC and the CVC.One of the biggest risks in the project is the battery. We have to make sure that all the charging components perform accordingly to the charging profile of the battery. We make sure this is happening by analyzing the values of the temperature sensors and the voltage sensors. To ensure safety we have cases in the code that turn off all the relays if there is a malfunction in the battery and or the circuit.

4 * Experiment Design and Feasibility Study

4.1 * Experiment Design

Min Hua Wu:

For the CVC, our simulation shows that the voltage across the two load resistors should be no more than 1.55V with +12V VCC+ and -12V VCC- as input voltage. The CVC utilizes a buck converter which steps down voltage from its input supply to its output, and a couple of relays to ensure that the circuits do not interact with each other (eg. both circuits cannot be on at the same time). The simulation shows that while giving the circuit a +-12 V input, the voltage across the load resistors is around 1.53V.

Gilberto Peraza-Martinez :

CVC

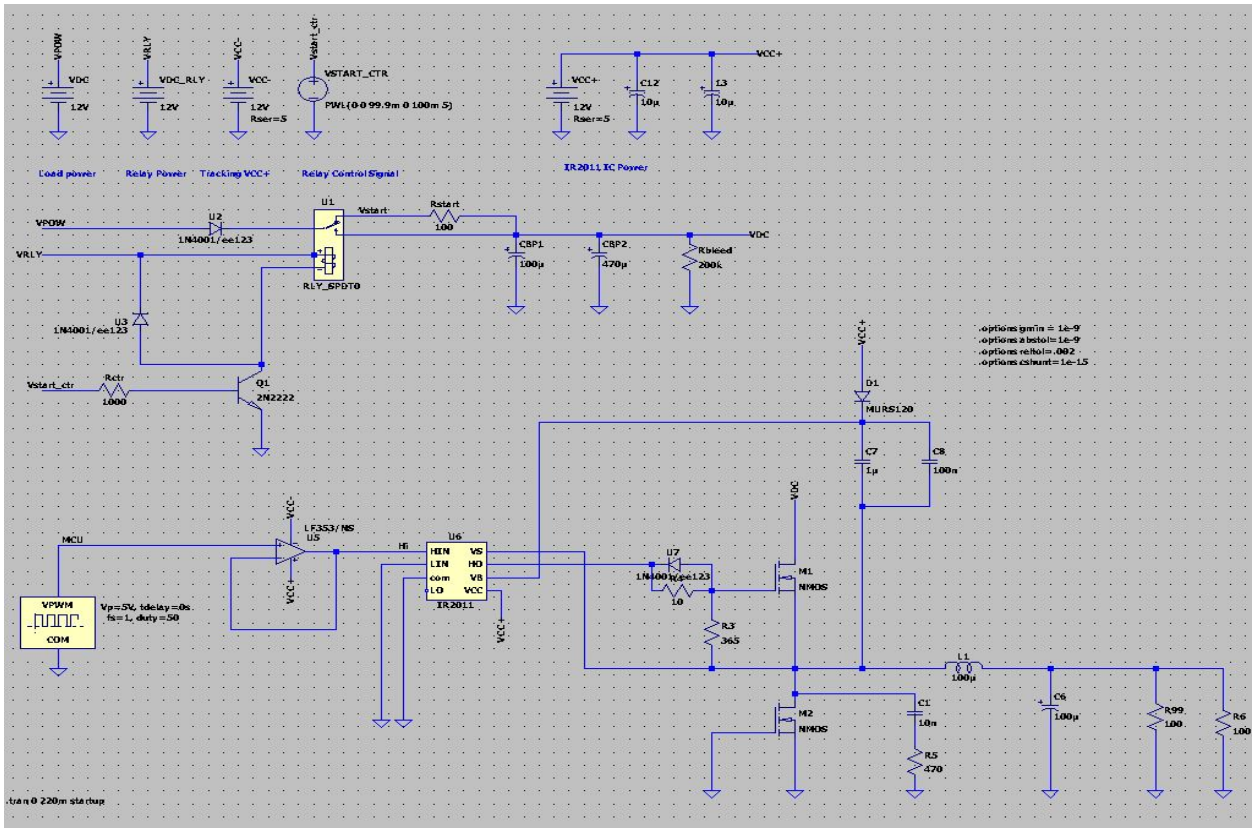
When designing the constant voltage circuit there was a particular resistor ,R3 that was instrumental to ensuring that the CVC circuit actually worked without supplying the load with an excessively high voltage. The process of finding the appropriate resistor was done in LTspice for the safety of myself and my team members. This precise value was very important to the circuit operation and as such we had to use multiple resistors to get this exact value. In real life we chose lower voltages to test the circuit at and increased the voltage once we were sure that the circuit was maintaining a safe temperature.

CCC

When designing the constant current circuit we used an initial 12 Volts in LTspice and used a 100K resistor. We wanted to know what kind of resistor would be required to reach 1/10C. We used LTspice to run these experiments.

DC

The discharge circuit was built before being tested on LTspice but we expected it to work well anyways. We attached a 10 Ohm resistor and a capacitor in parallel to the battery and observed times for discharge.



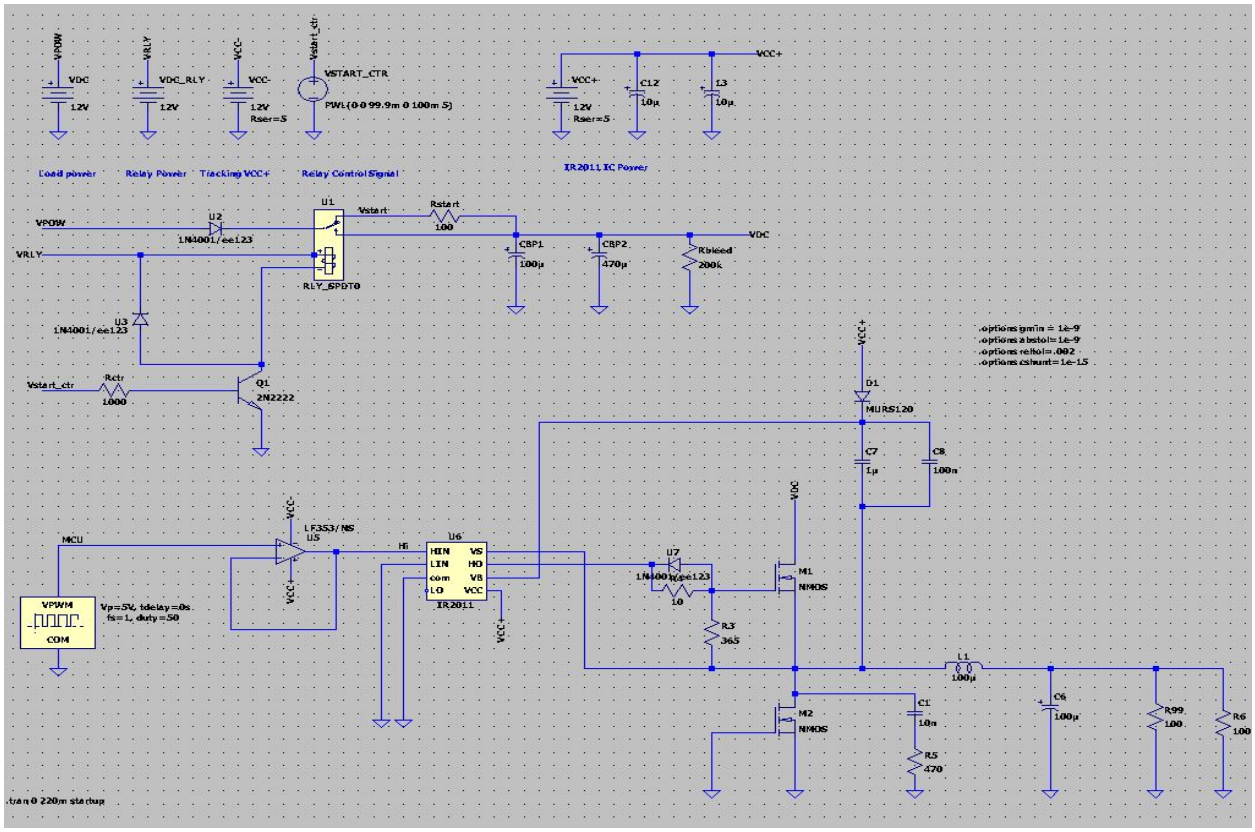
CVC is completed by Min Hua Wu, CCC is completed by Gilberto Peraza-Martinez, MCU is completed by Rohan Badiga

4.2 * Experiment Results, Data Analysis and Feasibility

Gilberto Peraza-Martinez

CVC

When I ran the CVC, there was a high amount of power being dissipated by the load and there was a large swing in voltage being shown at the load. Varying from ~4V and ~11 V was a problem which I was able to solve by applying a signal to the op amp in the bottom left of the CVC.



This was a simple fix but the high dissipation of wattage was still troublesome. There was not expected when running the circuit in real life but by unit testing every piece of hardware and the voltage at multiple points besides the load we came to the conclusion that we were simply going to have to add multiple resistors of lower values to reduce the power being dissipated at the load. This was done with 4 100Ohms resistors and a singular 20 ohm resistor before reaching the load. This allowed for the load to experience voltages around 1.53V with a variance of .65% (around .01 V)

CCC

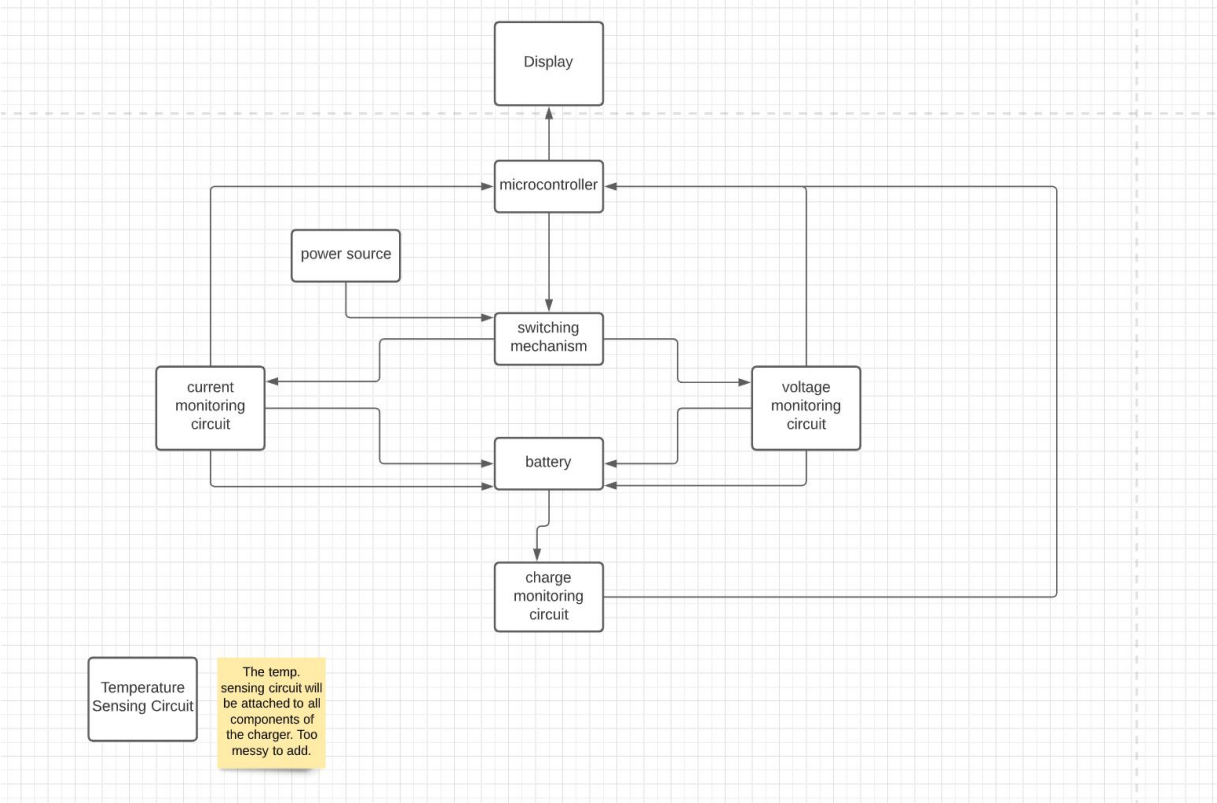
The constant current circuit was relatively simple unlike the CVC but even then we had issues. Comparing the results of the LTspice simulation was not necessary because the circuit was not dependably operational with the 2N3055 transistors and as such they were switched out for 2N222's which were chosen for their smaller size, dependable electrical connectivity and the max collector current rating of 800mA.With this kind of transistor we were able to reliably run the circuit with in 4%percent of the expected values of the simulation values containing the 2N222's (Simulation values were 100mA and in real life they were 97mA).

DC

The discharge circuit was reliable and did not discharge the battery at an exceedingly fast pace.

5 * Architecture and High Level Design

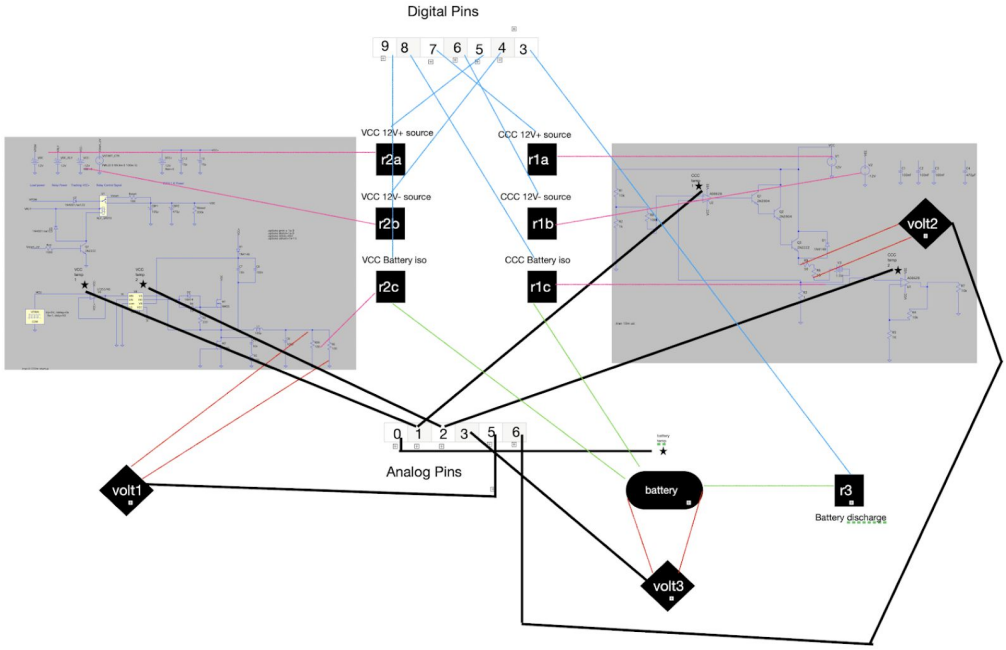
5.1 * System Architecture and Design



5.2 * Hardware Architecture

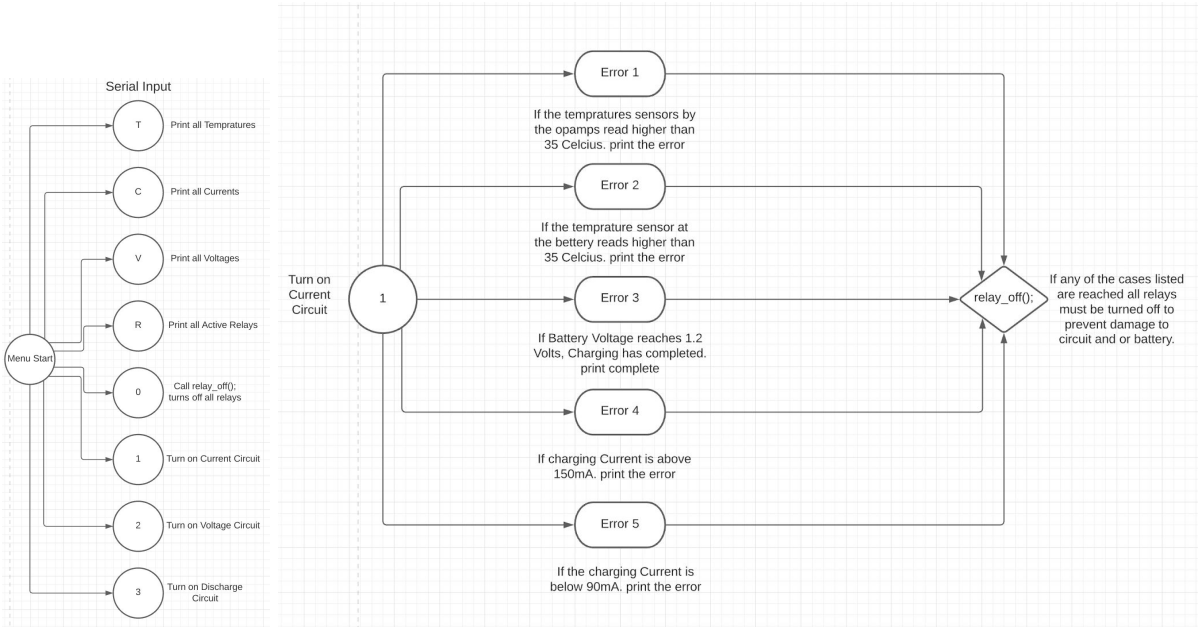
Gilberto Peraza-Martinez is responsible for the constant current circuit, and Min Hua Wu is responsible for the constant voltage circuit. Rohan Badiga was responsible for the Microcontroller Integration to control the circuits and have all peripherals active to read and log data.

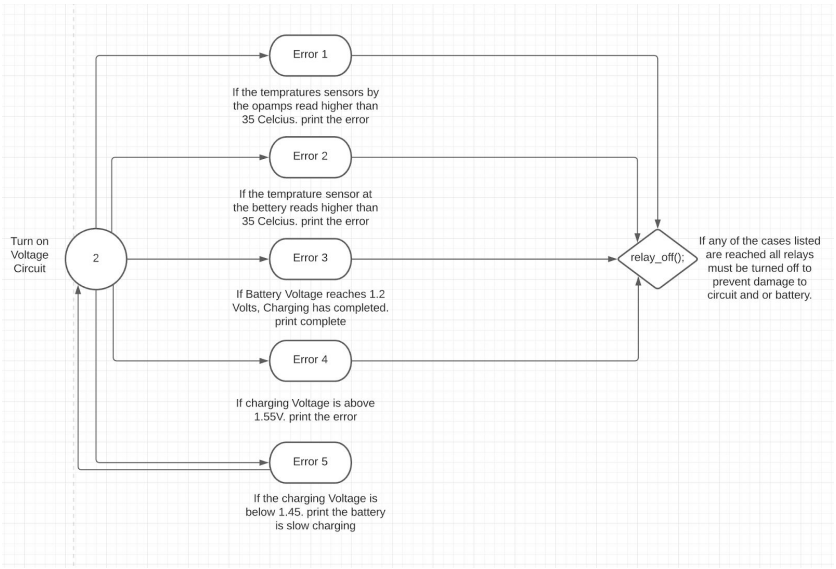
Simulations for Constant Voltage Circuit were made by Gilberto Peraza-Martinez and Min Hua Wu. Simulations for Constant Current Circuit were made by Rohan Badiga



5.3 * Software Architecture (only required if your design includes software)

Rohan Badiga is responsible for all Software Development, Implementation and Integration





5.4 * Rationale and Alternatives

The reason we chose this hardware setup with the relays is to ensure complete electrical isolation. This ensures when one circuit is turned off no residual current is traveline between circuits and/or to the battery. In addition the utilization of this many sensors lets us know all the hot spots in the circuit. During testing we had noticed the op-amps increasing in temperature and it is very important that they don't over overheat and damage the circuit or the battery. Possible alternatives to implementation would include modifying the relays so that two circuits can be connected to one relay. This would simplify the design. However, for the prototype we wanted to be as safe as possible.

6 Data Structures (include if used)

6.1 Internal software data structure

Rohan Badiga is incharge of all software implementation.

Digital Pins of Arduino

int r1a =7; // This identified the pin for the relay that controlled the 12V+ power supply for the CCC.

int r1b=6; // This identified the pin for the relay that controlled the 12V- power supply for the CCC.

int r1c= 8; // This identified the pin for the relay that isolated the CCC from the battery.

int r2a = 5; // This identified the pin for the relay that controlled the 12V+ power supply for the CVC.

int r2b=4; // This identified the pin for the relay that controlled the 12V- power supply for the CVC.

int r2c = 9; // This identified the pin for the relay that isolated the CVC from the battery.

int r3=3; // This identified the pin for the relay that isolated the Discharge Circuit from the battery.

Analog Pins of Arduino

float temp_sensor_battery=A0; // This identifies the analog input for the temperature sensor at the battery. Uses function float Read_temp_battery(); to calculate.

float temp_sensor2 = A1; // This identifies the analog input for the temperature sensors at op-amp1 in CCC and CVC. Uses function float Read_temp1(); to calculate.

float temp_sensor1 =A2; // This identifies the analog input for the temperature sensors at op-amp2 in CCC and CVC. Uses function float Read_temp2(); to calculate.

float voltage_sensor2=A3; // This identifies the analog input for the voltage sensor at the battery. Using float voltagefunc_sensor2(); function we can calculate the voltage value at the battery at any given moment.

float current_sensor=A4; // This identifies the analog input for the voltage sensors at the entry point for the battery in the CCC. Using float current_sensorf(); function we take that voltage value and identify the current going into the battery at any given moment.

float voltage_sensor1=A5; // This identifies the analog input for the voltage sensors at the entry point for the battery in the CVC. Using float voltagefunc_sensor1(); function we can calculate the voltage value going into the battery at any given moment.

6.2 Global data structure

Rohan Badiga is in charge of all software implementation.

void key_display(); // This function would keep running the background. Any user input from the menu would result in that action being called.

float voltagefunc_sensor1(); // This function used the voltage sensor in the CVC to calculate the amount of voltage that is being sent to the battery.

float voltagefunc_sensor2(); // This function used the voltage sensor at the battery to calculate and measure the voltage level of the battery so that we know how much the battery has charged.

float current_sensorf(); // This function uses the voltage sensor in the CCC to calculate and measure the current that is being sent into the battery.

float Read_temp_battery(); // This function calculates the temperature being read from the sensor at the battery.

float Read_temp1(); // This function calculates the temperature being read from the sensor at op-amp1 from either the CVC or the CCC depending on which one is turned on.

float Read_temp2(); // This function calculates the temperature being read from the sensor at op-amp2 from either the CVC or the CCC depending on which one is turned on.

float read_realay_state(); // This function sees what relays are currently turned on and lets us know what circuit is live and operational

void loop(); // This is the main function that has all the above functions running simultaneously. It lets the user know whenever there is an issue.

void relay_off(); // This function turns off all the relays and is called whenever there is an error that could damage either the circuit or the battery. It is also called when the battery has completed charging.

void setup(); // This function initializes all the relays that are being used so that they can be activated or deactivated accordingly. This function is also where the Menu is printed in the beginning of the start of the project.

6.3 Temporary data structure

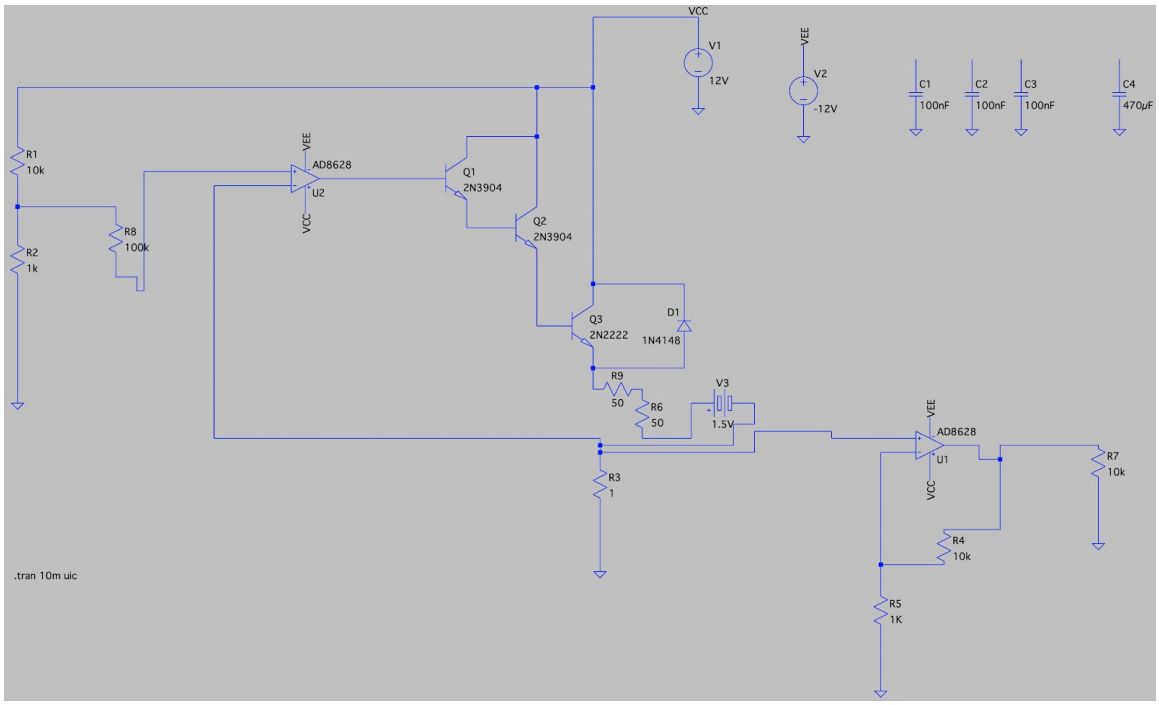
No temporary Files were used

6.4 Database descriptions

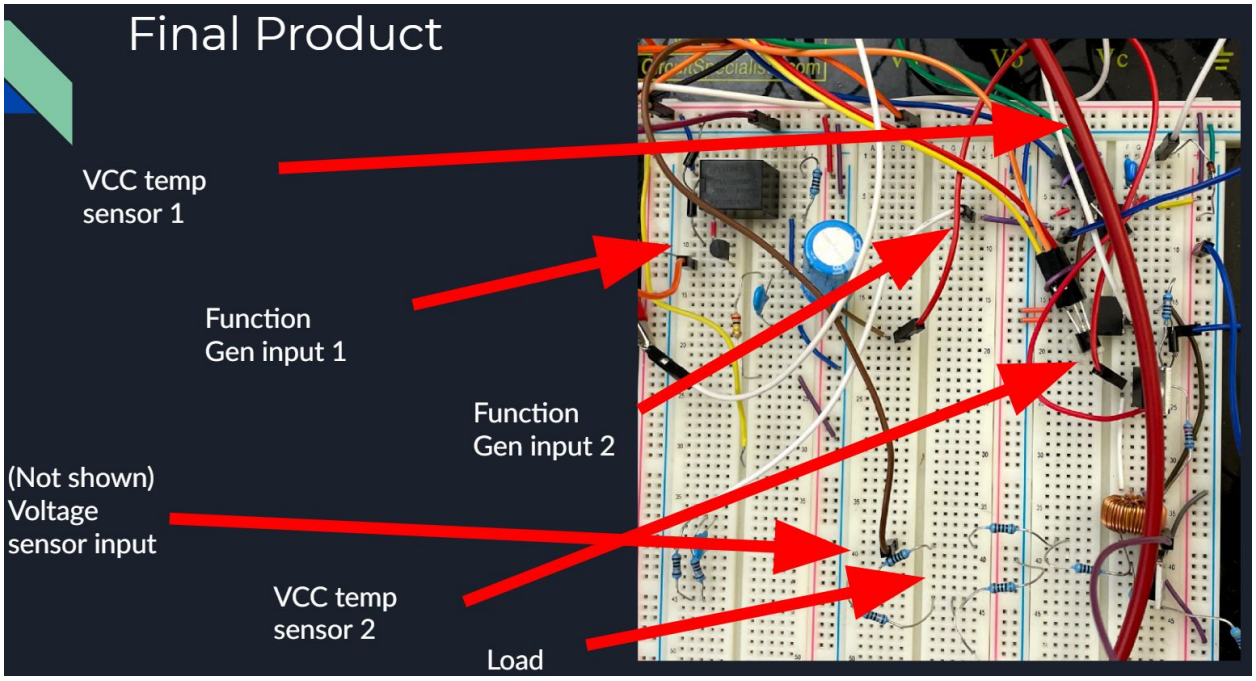
No Databases were used.

7 * Low Level Design

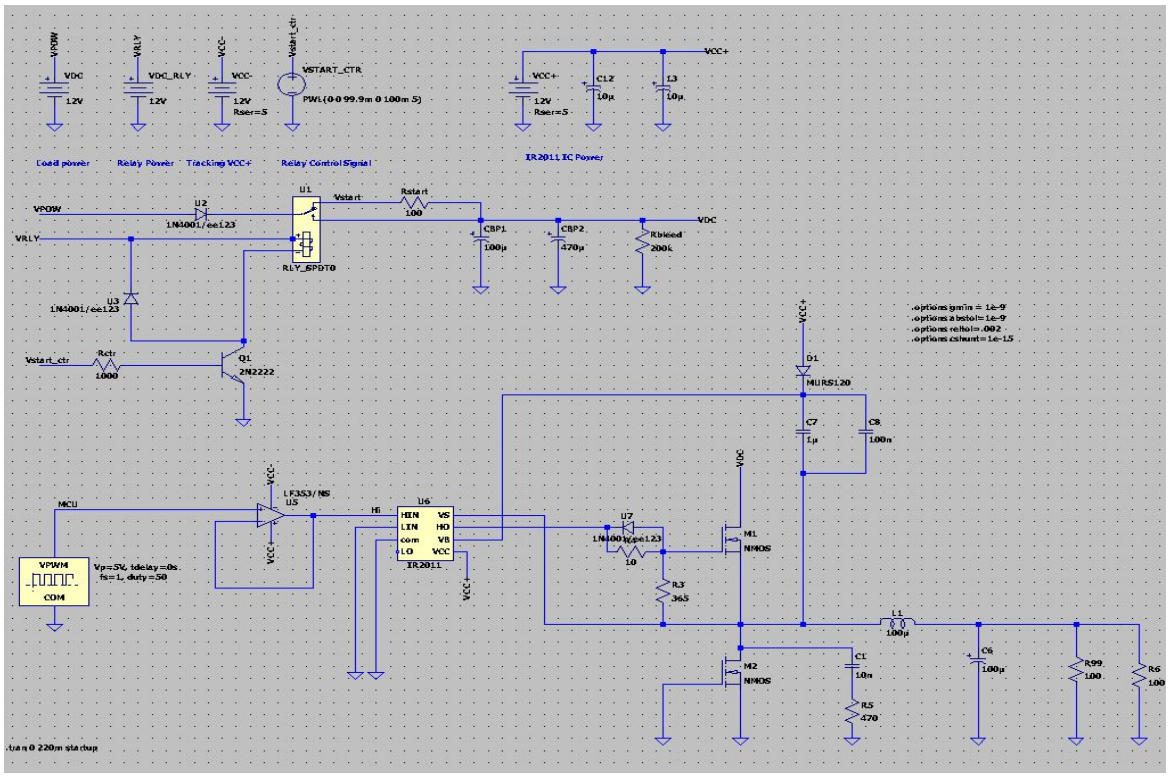
Constant Current Circuit (CCC):



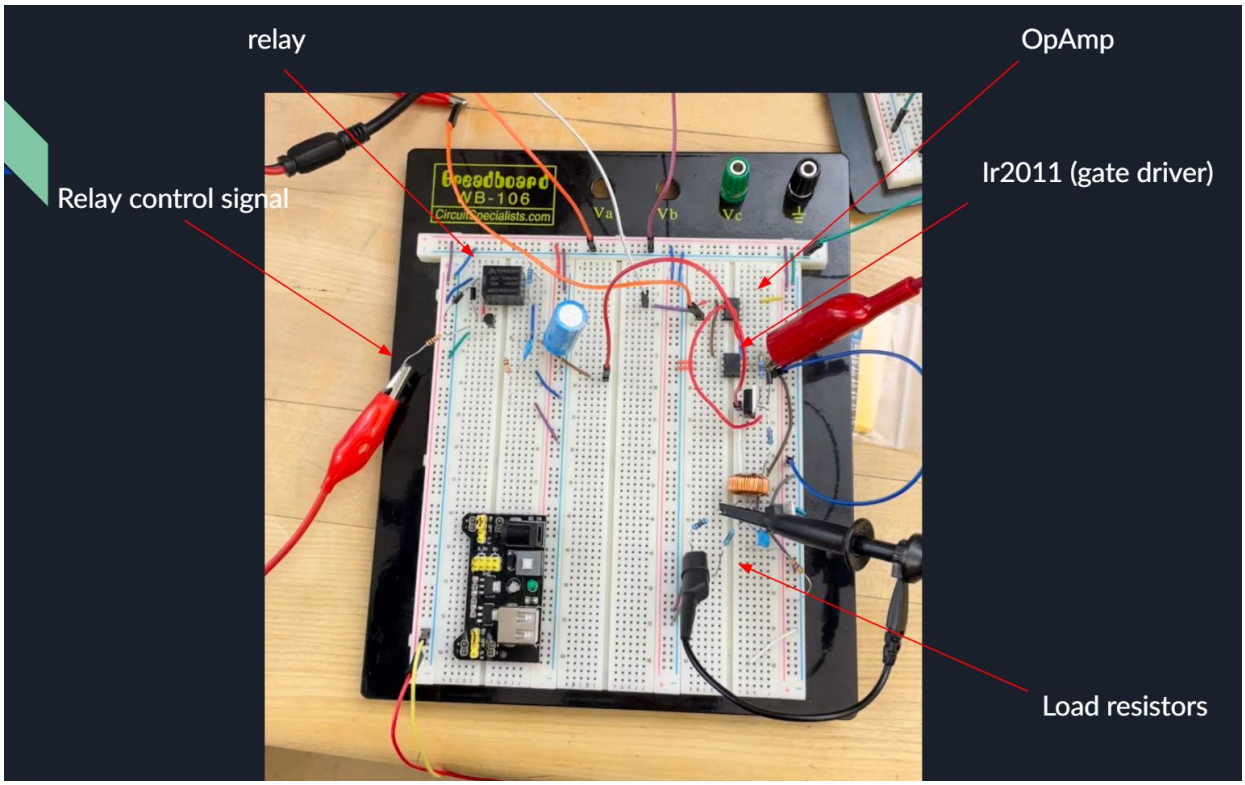
Actual CCC:



Constant Voltage Circuit (CVC):

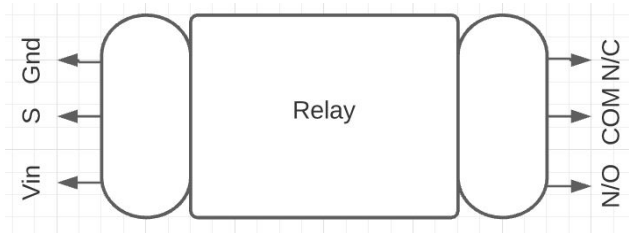

VPWM
VRLY
VCC-
Vstart_ctr
VDC
VDC_RLY
VSTART_CTR
PW(-0 0 99.9m 0 100m 5)
VCC+
C12
10µ
L3
10µ
12V
Rser=5
Load power
Relay Power
Tracking VCC+
Relay Control Signal
IR2011 IC Power
U1
Vstart
Rstart
100
CBP1
100µ
CBP2
470µ
Rbleed
200k
RLY_SPDT0
U2
1N4001/eee123
U3
1N4001/eee123
VRLY
Rctr
1000
Q1
2N2222
D1
MURS120
C7
1µ
C8
100n
.options gmin = 1e-9
.options abstol=1e-9
.options reltol=.002
.options cshunt=1e-15
MCU
LF353/NS
U5
VCC-
VCC+
Hi
U6
HIN
LIN
com
LO
VS
HO
VB
VCC
IR2011
U7
1N4001/eee123
10
M1
NMOS
R3
365
VPWM
COM
Vp=5V, tdelay=0s
fs=1, duty=50
L1
100µ
M2
NMOS
C1
10n
R5
470
C6
100µ
R99
100
R6
100
.tran 0 220m startup

Actual CVC:


relay
OpAmp
Relay control signal
Ir2011 (gate driver)
Load resistors
Breadboard
WB-106
CircuitSpecialists.com
Va
Vb
Vc

7.1 *Relays* (JQC - 3FF)

We used 7 relays controlled by the MCU. 3 worked with the CCC, 3 worked with the CVC and 1 worked with the Discharge circuit.



Rohan Badiga is in charge of implementation.

7.1.1 Processing narrative for *Relays*

First we needed to understand how the relay operated. we plugged it into the breadboard and had the N/O channel connected to a 5V+ power source. Then we proceeded to attach a resistor and a LED to the COM channel. Then we connected the Arduino to the S pin and the 5V+ power supply and ground to the corresponding pins. We initialized the Arduino pin to the relay and tested the code to operate the relay to turn the LED on and off. After we slowly integrated the relays into each circuit.

7.1.2 *Relays* interface description

Relay 1a is connected to the 12V+ power supply for the CCC. Relay 1b is connected to the 12V power supply for the CCC. Relay 1c is connected to the load resistors in the CCC and to the battery in series as a measure of electrical isolation.

Relay 2a is connected to the 12V+ power supply for the CVC. Relay 2b is connected to the 12V power supply for the CVC. Relay 2c is connected to the load resistors in the CVC and to the battery in parallel as a measure of electrical isolation.

Relay 3 is connected in series to a resistor and capacitor as the Discharge Circuit and to the battery as a measure of electrical isolation.

7.1.3 *Relays* processing details

1. This is the Initialization step
    a. pinMode(r1a,OUTPUT);

       pinMode(r1b,OUTPUT);

       pinMode(r1c,OUTPUT);

       pinMode(r2a,OUTPUT);

       pinMode(r2b,OUTPUT);

       pinMode(r2c,OUTPUT);

```
pinMode(r3,OUTPUT);
```

2. This is the step used to turn all the relays off
   a.
```
void relay_off()
{
digitalWrite(r1a,LOW);
digitalWrite(r1b,LOW);
digitalWrite(r1c,LOW);
digitalWrite(r2a,LOW);
digitalWrite(r2b,LOW);
digitalWrite(r2c,LOW);
digitalWrite(r3,LOW);
int x = 0;
}
```

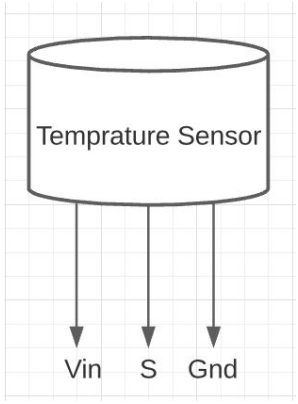
3. This is the step for all each of the relays being turned on depending on the user input in the menu
   a.
```
if (inChar == '0') {
relay_off();
Serial.println("All relays turned off");
}
  if (inChar == '2') {
    digitalWrite(r1a,LOW);
    digitalWrite(r1b,LOW);
    digitalWrite(r1c,LOW);
    digitalWrite(r3,LOW);
    digitalWrite(r2a,HIGH);
    digitalWrite(r2b,HIGH);
    digitalWrite(r2c,HIGH);
     x = 2;
    Serial.println("Voltage Circuit Turned On");
  }
  if (inChar == '1') {
    digitalWrite(r1a,HIGH);
    digitalWrite(r1b,HIGH);
```

```
  digitalWrite(r1c,HIGH);
  digitalWrite(r3,LOW);
  digitalWrite(r2a,LOW);
  digitalWrite(r2b,LOW);
  digitalWrite(r2c,LOW);
   x = 1;
  Serial.println("Current Circuit Turned On");
 }
 if (inChar == '3') {
  digitalWrite(r1a,LOW);
  digitalWrite(r1b,LOW);
  digitalWrite(r1c,LOW);
  digitalWrite(r3,HIGH);
  digitalWrite(r2a,LOW);
  digitalWrite(r2b,LOW);
  digitalWrite(r2c,LOW);
  x = 3;
  Serial.println("Discharge Circuit Turned On");
 }
```

Overall the implementation of all the relays worked perfectly. One of the design restraints that we faced included the powering of all the relays. We used the 5V power supply from the Arduino to power all of them but it turned out that the current coming from the Arduino was not sufficient in powering all of them at once. To address this we added more power supplies.

7.2 *Temperature Sensor* (LM 35)

We used 5 Temperature sensors controlled by the MCU. 2 located at each op-amp in the CCC, 2 located at each op-amp in the CVC, and 1 located at the Battery.



Rohan Badiga is incharge of implementation.

7.2.1 Processing narrative for *Temperature Sensor*

Plugging the temperature sensor into the Arduino we would be given an analog value dependent on the voltage that the sensor is emitting to the Arduino. Then we would convert that value into a comprehensible value into Celcius. We tested our values with an actual thermometer. We were within 2% accuracy.

7.2.2 *Temperature Sensor* interface description

There are two temperature sensors located in the CCC and the CVC. Each being at an op-amp, they measure the ongoing temperature when each one is being actively used. The reason we put them at each op-amp is because in previous circuit tests we noticed that these were the spots generally in the circuit that would heat up. We also placed one right on the battery to know if the battery is having a bad reaction to the charging. The placement of these sensors lets us know if there is any issue with the performance of the circuits and it also helps us protect the circuits and battery in case something were to happen. The temperature sensor for the battery is connected directly into the Arduino. The temperature sensor at op-amp1 in CCC and CVC are plugged in series to the same analog pin. The same rule applies for the temperature sensors at op-amp 2 in the CCC and CVC. Since only one of the circuits will be on at a time the power supply for the sensors is powered by the same relay that powers the circuits. Only the sensors in the respective circuit turn on.

7.2.3 *Temperature Sensor* processing details

A detailed description for each module is presented, including hardware, algorithm, local data structures, design constraints, limitations, performance issues, etc.

1. This function uses the temperature sensor at the battery and calculates the temperature in Celsius
    a. float Read_temp_battery()

```
{
 int rawtempbattery = analogRead(temp_sensor_battery);
 float degree_battery = (rawtempbattery / 1024.0) * 5000;
 float celsius5 = degree_battery / 10;
 return(celsius5);
```

```
}
```

2. This function uses the temperature sensor at op-amp1 of the circuit that is turned on (CCC or CVC) and calculates the temperature in Celsius
    a. float Read_temp1()

```
{
 int rawvoltageccc1 = analogRead(temp_sensor1);
 float millivolts3 = (rawvoltage1 / 1024.0) * 5000;
 float celsius3 = millivolts3 / 10;
 return(celsius3);
}
```

3. This function uses the temperature sensor at op-amp2 of the circuit that is turned on (CCC or CVC) and calculates the temperature in Celsius
    a. float Read_temp2()

```
{
 int rawvoltagevcc2 = analogRead(temp_sensor2);
 float millivolts2 = (rawvoltage2 / 1024.0) * 5000;
 float celsius2 = millivolts2 / 10;
 return(celsius2);
 }
```

4. These cases are inside loop(); and print out an error whenever the temperature gets too hot either at the op-amps or the battery, and it also turns off the relays if that is the case.
    a. if (temp2 > 35)

```
{
 Serial.println(" opamp 1 too hot");
 relay_off();
 }
if (temp1 > 35)
{
 Serial.println(" opamp 2 too hot");
  relay_off();
}
if (bat_temp > 35)
{
 Serial.println(" battery too hot");
 relay_off();
```

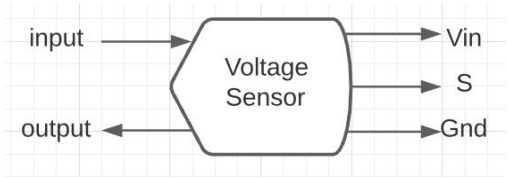
```
}
```

5. This case is in the key_display(); function. Whenever the user input the letter 'T' all the live temperatures are printed out.
    a.
```
if (inChar == 'T') {
    float tep1 =Read_temp1();
    float tep2 = Read_temp2();
    float tep3 = Read_temp_battery();
     Serial.println(tep1);
     Serial.println(tep2);
     Serial.println(tep3);
    }
```
6. Formulas used
    a. Vout = Vref * (reading from ADC/ 1024)
        i. Vref = 5000
    b. Temperature (°C) = Vout / 10

We originally faced a lot of issues trying to figure out how to utilize this many temperature sensors due to hardware limitations in the Arduino and limited amount of analog pins. But we addressed it by keeping 2 sensors in series into one pin and powering the sensors with the same relays that powered their respective circuits.

7.3 *Voltage Sensor (DC0-25V Voltage Tester)*

We used 2 Voltage sensors controlled by the MCU. 1 worked with the CVC to measure the amount of voltage that would be going into the battery, and 1 was used at the battery to measure its voltage to decipher the amount of charge it has.



Rohan Badiga is incharge of implementation.

7.3.1 Processing narrative for *Voltage Sensor*

We had to understand how the sensor operated first. Once we did that we tested our results with a power supply to see how accurate our measurements were. We were within 5% accuracy. Afterwards, we had to understand our implementation requirements and apply them to the code.

7.3.2 *Voltage Sensor* interface description

One voltage sensor was placed in parallel with the battery. This would give us the voltage of the battery and let us know how much it has charged. The second voltage sensor was placed in the CVC. This was placed in parallel to the load resistor so that we could see the voltage that is traveling to the battey.

7.3.3 *Voltage Sensor* processing details

A detailed description for each module is presented, including hardware, algorithm, local data structures, design constraints, limitations, performance issues, etc.

1. This function monitors the rate at which the CVC is actively charging the Battery. In our prototype we used a Nickel Cadmium battery which does not need to be charged with a CVC so we have our parameters set in the code as if it were to charge a Lithium Ion battery. However since we did not get that far into testing we tested these values over the resistors not actually connecting the CVC circuit to the battery. It turns off the circuit if the charging voltage gets too high and prints out a warning if the circuit is charging the battery at too low of a rate.
    a. float voltagefunc_sensor1()

```
{
 float volt_value1=analogRead(voltage_sensor1);
 float volt_valueraw1 = (volt_value1/1010)*25;
 if (volt_valueraw1>1.55)
 {
  relay_off();
 Serial.println("Charging Voltage too high");
 Serial.println(volt_valueraw1);
 }
 if (volt_valueraw1==1.45)
 {
   Serial.println("Battery Charging Low");
 }
 else
 return(volt_valueraw1);
}
```

2. This function lets us know the voltage at the battery and shuts off the circuit if the battery has been completely charged
    a. float voltagefunc_sensor2()

```
{
 float volt_value3=analogRead(voltage_sensor2);
 float volt_valueraw2 = (volt_value3/1010)*25;
  if (volt_valueraw2==1.2)
```

```
{
  relay_off();
  Serial.println("Battery Fully Charged");
}
else
 return(volt_valueraw2);
}
```

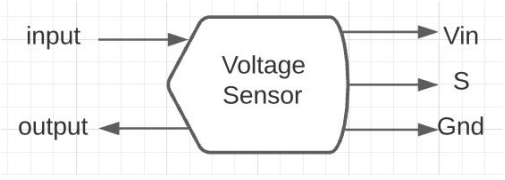
3. This case is in the key_display(); function. Whenever the user input the letter 'V' all the live voltages are printed out.
    a.
```
if(inChar == 'V'){
  float volts1 = analogRead(voltage_sensor1);
  Serial.println(volts1);
  float volts2 =analogRead(voltage_sensor2);
  float vbat= (volts2/1000)*25;
  Serial.println(vbat);
  }
```

4. Formulas Used
    a. analogVoltage = (analogVoltage/maxanalogValue)*sensormaxVoltage;
        i. maxanalogValue = 1010; // highest integer given at max input voltage
        ii. sensormaxVoltage = 25; //highest input voltage of sensor being used
        iii. analogVoltage is what's being read from the pin

7.4 *Voltage Sensor as a Current Sensor (DC0-25V Voltage Tester)*

Provide or reference a detailed description, schematic and/or diagrams of this module. Repeat this section for each module i.

Aside from the 2 voltage sensors mentioned above we used 1 more voltage sensor as a current sensor in the CCC controlled by the MCU to identify the charging current going into the battery.



Rohan Badiga is incharge of implementation.

7.4.1 Processing narrative for *Voltage Sensor as a Current Sensor*

We tried out multiple Current sensors that we had purchased, but none were able to perform to our expectations so we decided to use one of the voltage sensors and calculate the current over the load resistors. To test this we tested what the current would be in our circuit with a Multimeter and then we compared it to the value that we were getting from this sensor. It was within 2% accuracy.

7.4.2 *Voltage Sensor as a Current Sensor* interface description

We needed a way to measure the current that is being sent to the battery by the CCC. To do this we implemented the voltage sensor in parallel over the load resistors. Since we already knew the resistance of the resistors, we used it and the voltage at that point to calculate the Current feeding into the battery.

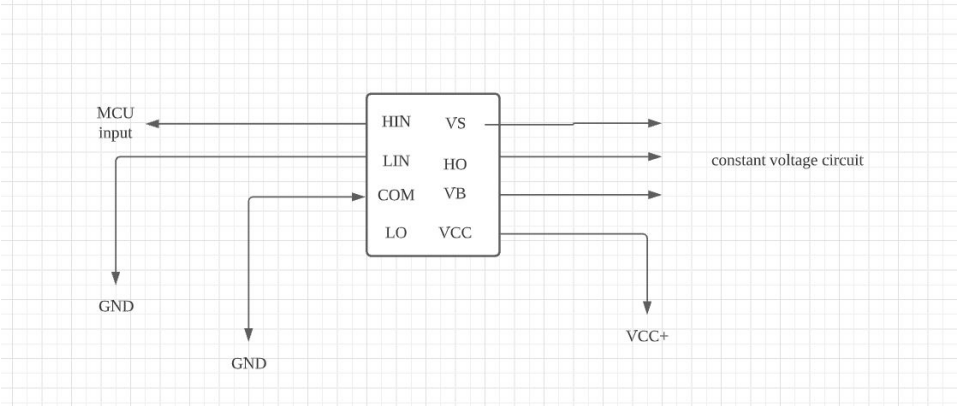
7.4.3 *Voltage Sensor as a Current Sensor* processing details

1. This function calculates the current over the load resistors and checks to see if the current going to the battery is too high or too low and prints the error while turning off the circuit to prevent harm to battery or circuit.
    a.
```
float current_sensorf()
{
float current = analogRead(current_sensor);
 float v= .0048828125 * current;
 v -= 2.5;
 float amps = v / .066;
 amps = current;
 if (current > .15){
  relay_off();
  Serial.println("current too high");
  Serial.println(current);
 }
 if (current < .09) {
  relay_off();
 Serial.println("current low");
 Serial.println(current);
 }
 return (current);
}
```
2. This case is in the key_display(); function. Whenever the user input the letter ‘C’ the current over the load resistor gets printed out.
    a.
```
if (inChar == 'C') {
    float volts3 =analogRead(voltage_sensor1);
```

```
 float vres= (volts3/1000)*25;
float curr1 = vres/100;
Serial.println(curr1, 3);
}
```

3. Formulas used
    a. voltage/resistance = current
    b. resistance = 100 Ohms

7.5 *IR2011 Gate Driver*



Min Hua Wu is in charge of the implementation of the IR2011 Gate Driver.

7.5.1 Processing narrative for *IR2011*

The MCU is connected to the logic input for the high side gate driver (HIN), and the high side gate drive output (HO) is connected to a buck converter, which steps down the voltage from the high side floating supply (VB).

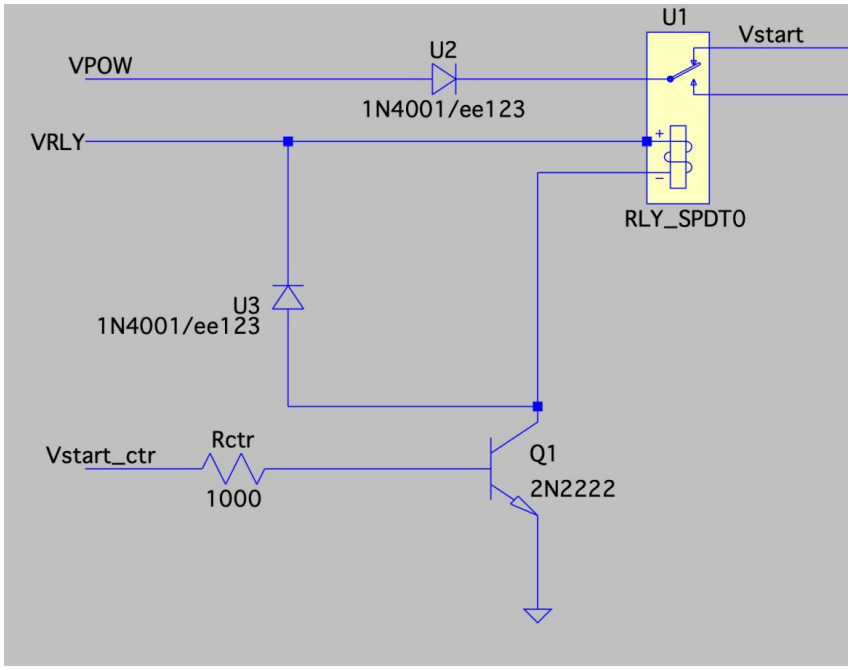
7.5.2 *IR2011* interface description

The MCU input goes through an LM358 amplifier, the high side floating supply (VB), high side gate drive output (HO), and high side floating supply return (VS) are connected to the rest of the CVC (Refer to CVC diagram above). The low side supply (VCC) is connected to the +12V input while the logic input for low side gate driver (LIN) and low side return (COM) both connect to circuit ground.

7.5.3 *IR2011* processing details

The IR2011 is a high power, high speed power MOSFET driver that has independent high and low side referenced output channels. In our design we only connected the IR2011 using the high side of the channel, and the high side gate drive output produces a high-current drive input for the gate of a high power MOSFET which in our case is the 2N3904.

7.6 *JQC-T78 Relay*



Min Hua Wu is in charge of the JQC-T78 Relay.

7.6.1 Processing narrative for JQC-T78 *Relays*

The positive and negative end of the coil of the Single Pole Double Throw (SPDT) relay is connected to our relay power (VRLY) and relay control signal (Vstart_ctr). The relay start control controls if load power (VPOW) is connected to the normally closed or normally open port.

7.6.2 *JQC-T78 Relay* interface description

The relay power (VRLY) powers the relay and relay control signal (Vstart_ctr) controls whether load power (VPOW) is connected to the normally open (NO) or normally close (NC). Vstart is connected to a 100 ohm resistor which acts as an inrush current limiter. The right side of the relay is connected to the rest of the CVC to ensure that the load power is stepped down in the buck converter connected to the IR2011.

7.6.3 *JQC-T78 Relay* processing details

The JQC-T78 Relay is simply a switch that allows load power to be properly stepped down by the connected buck converter. In our simulation the relay action was emulated by manually setting the Vstart_ctr to turn on to 5V at 100ms. Then we connect VDC (which is on the right side of the relay) to the drain of the high-side MOSFET M1.

8 * Technical Problem Solving

8.1 * The Short Circuit Ground Problem

While testing the CCC for 5 minutes, one of the resistors started to smoke and burn until we quickly turned off the power supply. (Min Hua and Gilberto)

8.2 * Solving the Short Circuit Ground Problem

After we turned off the power supply, we identified that the common ground connecting to the power supply was not connected to the circuit ground on the breadboard. And after we swapped out the burnt resistor with a new one and connected the power supply ground to the circuit. The circuit was working again after.

8.3 * The Not Enough Current for Relays Problem

In order to control the relays, we were using the kit we got from EECS120B class and turns out the board we had only supplied a 5V power, and with the amount of relays we had, we could not control them all at the same time. (Rohan)

8.4 * Solving the Not Enough Current for Relays Problem

In order to resolve this issue, we had to utilize multiple 5V power sources to power all the peripherals in our project.

8.5 * The Current Sensor Problem

For the CCC we need a current sensor to monitor the overall output of the current before going into the battery. We had tried multiple different current sensors. Either they did not work or the data that they provided was far too inaccurate. (Rohan, Gilberto)

8.6 * Solving the Current Sensor Problem

In order to resolve this issue, we utilized a voltage sensor to calculate the current with the given resistance it was measuring over.

8.7 * The LT Spice Problem

When we were making the simulation for the CCC, we would initially get a current range of 1.05 micro Amps for charging the battery. (Rohan, Gilberto)

8.8 * Solving the LT Spice Problem

In order to resolve this issue, we had to research different ways to change the circuit in order to make the output current 100 milli Amps and above. We added and modified the resistors to attain the required output current.

8.9 * The Burning op-amp Problem

In the early stages of development for the CCC, one of the op-amps was getting hot and started smoking. Min Hua and Gilberto identified the issue in the board and fixed it. We needed a way to identify if things were to get hot again. (Rohan, Min Hua)

8.10 * Solving the Burning op-amp Problem

We fixed the problem by connecting the op-amp to the negative power source rather than the ground we had it connected to initially. To ensure we would be aware of high temperature op-amps in the future, we implemented temperature sensors into both the CCC and the CVC circuit wherever there was an op-amp. This way we can identify where the problem is before something catches fire.

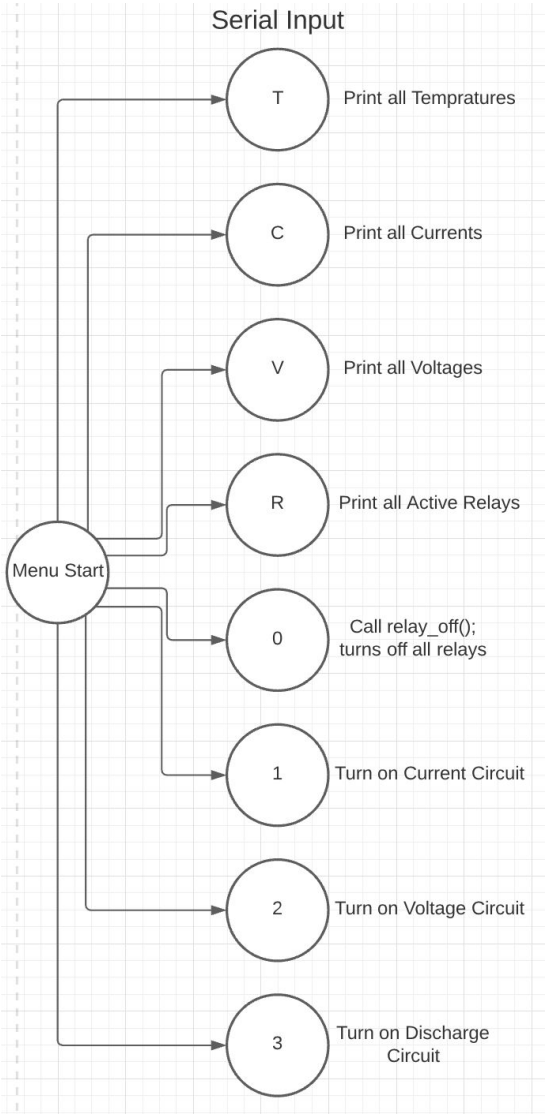
9 User Interface Design

Rohan is incharge of the User Interface Design

9.1 Application Control

Code:

```
Serial.println("Enter T for Temperature");
Serial.println("Enter C for Current");
Serial.println("Enter V for Voltage");
Serial.println("Enter R for Relay State");
Serial.println("Enter O for Turn off all Relay");
Serial.println("Enter 1 for Current");
Serial.println("Enter 2 for Voltage");
Serial.println("Enter 3 for Discharge");
```



9.2 User Interface Screens

```
void key_display(){
 while (Serial.available()) {
   char inChar = (char)Serial.read();
   if (inChar == 'T') {
    float tep1 =Read_temp1();
    float tep2 = Read_temp2();
    float tep3 = Read_temp_battery();

     Serial.println(tep1);
     Serial.println(tep2);
     Serial.println(tep3);
   }
   if(inChar == 'V'){
     float volts1 = analogRead(voltage_sensor1);
     Serial.println(volts1);

     float volts2 =analogRead(voltage_sensor2);
     float vbat= (volts2/1000)*25;
     Serial.println(vbat);
   }


    if (inChar == 'R') {

    if ( x == 1 ){
     Serial.println("Current opperational");
```

```
} else if ( x == 2 ){
 Serial.println("Voltage opperational");
}else if (x == 3 ){
 Serial.println("Discharge opperational");
} else if (x == 0 ){
 Serial.println("Everything is turned off");
}
}
if (inChar == 'C') {

 float volts3 =analogRead(voltage_sensor1);
 float vres= (volts3/1000)*25;
float curr1 = vres/100;
Serial.println(vres, 3);
Serial.println(curr1, 3);
}
if (inChar == '0') {
 relay_off();
 Serial.println("All relays turned off");
}
if (inChar == '2') {
  digitalWrite(r1a,LOW);
  digitalWrite(r1b,LOW);
  digitalWrite(r1c,LOW);
  digitalWrite(r3,LOW);
  digitalWrite(r2a,HIGH);
  digitalWrite(r2b,HIGH);
  digitalWrite(r2c,HIGH);
   x = 2;
  Serial.println("Voltage Circuit Turned On");
}
if (inChar == '1') {
  digitalWrite(r1a,HIGH);
  digitalWrite(r1b,HIGH);
  digitalWrite(r1c,HIGH);
  digitalWrite(r3,LOW);
  digitalWrite(r2a,LOW);
  digitalWrite(r2b,LOW);
  digitalWrite(r2c,LOW);
   x = 1;
  Serial.println("Current Circuit Turned On");
}
if (inChar == '3') {
  digitalWrite(r1a,LOW);
  digitalWrite(r1b,LOW);
  digitalWrite(r1c,LOW);
  digitalWrite(r3,HIGH);
  digitalWrite(r2a,LOW);
  digitalWrite(r2b,LOW);
```

```
      digitalWrite(r2c,LOW);
      x = 3;
      Serial.println("Discharge Circuit Turned On");
    }
  }
}
```

10 * Test Plan

10.1 * Test Design

Test Case 1 (CCC with Load Resistor) Gilberto & Min Hua

1. Objective: To measure the current going through the load resistors of the CCC
2. Experiment setup: We simulate the inputs of the CCC using the power supply, and measure the voltage drop across the load resistor, and we use the voltage and resistance values to calculate the current going through.
3. Experiment procedure: We use a multimeter to measure the voltage drop across the load resistor
4. Expected results: The current going through should be around 100mA

Test Case 2 (CCC with Battery) Gilberto & Min Hua

1. Objective: To use the CCC to charge a rechargeable nickel-cadmium battery
2. Experiment setup: Connect the battery in series with the load resistors of the CCC
3. Experiment procedure: First we measure the depleted battery voltage, then we let the battery charge for 5 hours, measure the battery voltage again to see if it increases.
4. Expected results: As long as the battery increases from the original value, the battery is being charged.

Test Case 3 (CVC with Load Resistor) Gilberto & Min Hua

1. Objective: To measure the voltage across the load resistors of the CVC
2. Experiment setup: We use a signal generator to control the relay control signal, and use power supply to give us the +-12V.
3. Experiment procedure: We use the multimeter to measure the voltage across the load resistors
4. Expected results: The simulation shows that we should not get more than 1.55V across the load resistors

Test Case 4 (CVC Relay) Gilberto & Min Hua

1. Objective: To test if the relay is working properly.
2. Experiment setup: Connect the relay like in the CVC schematic, then use a signal generator to simulate input function of square wave with 5V amplitude and 50% duty cycle over 10 second.
3. Experiment procedure: Connect the relay power and relay control signal, then relay will start
4. Expected results: The relay will click every 5 seconds to switch from normally open to normally close

Test Case 5 (Discharge with Battery) Gilberto, Min Hua, Rohan

1. Objective: To discharge a rechargeable battery so we can recharge the battery again.
2. Experiment setup: We used a resistor and capacitor to discharge the battery
3. Experiment procedure: Connect the resistor and capacitor in parallel with the battery.
4. Expected results: The battery voltage will drop from 1.2V since that is the battery voltage when it is fully charged.

Test Case 6 (Temperature Sensor) Rohan

1. Objective: To test the accuracy of the LM35 Temperature Sensor and to ensure that the code written for it is correct
2. Experiment setup: Holding temperature sensor and thermometer in one hand in a closed palm and comparing values after 10 seconds

3. Experiment procedure: We have the code output the temperatures from the sensors with a 10 second delay to give time to calibrate. The thermometer will be read after those 10 seconds as well
4. Expected result: The theoretical value will be the thermometer and the actual value will be the temperature sensor. The test is a success if the temperature sensor is within 2% of the thermometer.

Test Case 7 (Voltage Sensor) Rohan

1. Objective: To test to see if the value from the voltage sensor is accurate.
2. Experiment setup: Have a DC power supply at 5V, 10V, and 20V and Voltage sensor read it.
3. Experiment procedure: Have the code output read voltage from the voltage sensor. Compare it to the power supply voltage.
4. Expected Result: The Voltage sensor value must be within 10% of the actual value from power supply

Test Case 8 (Current Sensing) Rohan

1. Objective: To test to see if the voltage sensor that is used to calculate current is accurate.
2. Experiment setup: Have an active circuit, sensor connected to arduino with the code and a multimeter connected to measure for accuracy.
3. Experiment procedure: Have to code output the current that is calculated and compare it to the multimeter.
4. Expected Result: The current value must be within 10% of the value read by the multimeter

Test Case 9 (Relay) Rohan

1. Objective: To make sure relays are operational.
2. Experiment setup: Have relay connected to a power source and then to a LED.
3. Experiment procedure: We connect the Relay to the Arduino. Then the 5V power source to the N/O channel in the Relay. Then we have a resistor connected to an LED going to ground.
4. Expected Result: With the code you are able to turn the LED on and off.

10.2 * Bug Tracking

Any failures in the test cases will be dealt by the person in charge of that respective test.

10.3 * Quality Control

Test Case 1: Passed.

Test Case 2: Passed.

Test Case 3: Passed.

Test Case 4: Passed.

Test Case 5: Passed.

Test Case 6: Passed.

Test Case 7: Passed.

Test Case 8: Passed.

Test Case 9: Passed.

10.4 * Identification of critical components

Tests 1-5 are critical because they ensure that the project works.. They are the principal ways that the battery charges. Test 6-9 only deal with the User interface and peripherals to ensure the project runs smoothly.

10.5 * Items Not Tested by the Experiments

We have tested all the parts used within our schematics and circuits.

11 * Test Report

11.1 * Test 1

Gilberto Peraza, Min Hua Wu

1. The actual current we were getting was around 90mA.
2. The expected current was 100mA, so we are within the 10% range of error
3. Because simulation does not account for real life constraints such as resistor values being off or wire resistance, our result was a little smaller than expected.

11.2 * Test 2

Gilberto Peraza, Min Hua Wu

1. We were able to slowly bring the battery voltage from 1.04V to 1.19V after around 5 hours of charging
2. The battery voltage has been increasing since the time of charge, which is exactly what we want.
3. We were able to bring a battery that was discharged for an entire night back to close to the initial battery voltage.

11.3 * Test 3

Min Hua Wu, Gilberto Peraza

1. The voltage we were getting across the load resistor of the CVC is around 1.5V
2. The expected results show that we are not suppose to get over 1.55V
3. Pass

11.4 * Test 4

Min Hua Wu, Gilberto Peraza

1. The click was happening every 5 seconds
2. We heard the click every 5 seconds turning the relay on and off
3. The relay is working as we expected

11.5 * Test 5

Gilberto Peraza, Min Hua Wu, Rohan Badiga

1. Depleted Battery Voltage: 0.8V
2. Comparing to original battery voltage which is 1.2V, the battery has been discharged
3. The discharge circuit was able to discharge a battery

11.6 * Test 6

Rohan

1. Temperature Sensor
    a. Sensor: 95
    b. Sensor: 93
    c. Sensor: 96

   d. Sensor: 94
   e. Sensor: 93
2. Comparison with expected results
   a. Thermometer: 94
3. Within 2%
   a. Pass

11.7 * Test 7

Rohan

1. Voltage Sensor
   a. Sensor: 5.02 V
   b. Sensor: 4.98 V
   c. Sensor: 4.95 V
   d. Sensor: 5.07 V
   e. Sensor: 5.04 V
2. Comparison with expected results
   a. DC Power Supply: 5.01V
3. Within 10%
   a. Pass

11.8 * Test 8

Rohan

1. Current Sensor
   a. Sensor: .105 A
   b. Sensor: .095 A
   c. Sensor: .098 A
   d. Sensor: .106 A
   e. Sensor: .115 A
2. Comparison with expected results
   a. Multimeter: .100A
3. Within 10%
   a. Pass

11.9 * Test 9

Rohan

1. Relay
   a. Relay on - LED on
   b. Relay off - LED off
   c. Relay on - LED on
   d. Relay off - LED off
   e. Relay on - LED on
   f. Relay off - LED off
2. The LED is supposed to turn on when the Relay has turned on. You must hear an audible click.
3. Relay state and LED state match

a. Pass

12 * Conclusion and Future Work

12.1 * Conclusion

Gilberto Peraza-Martinez :

The constant current circuit was my main responsibility and it is functional. It does not however charge more than one cell at a time which was something that we as a group really wanted to implement. Despite it being functional I really would have wanted to speed up the charging to an optimum charge rate .5C instead of the .1C that we are currently charging at. The reason that it does not currently charge at the rate it is meant to is because we did not implement a method to vary the current entering the first transistor of the CCC, which is the safest option of increasing the current entering the load.

Min Hua Wu:

The battery charger consists of 3 main components: constant current charging circuit, constant voltage charging circuit, and microcontroller unit to switch between the circuits. Gilberto Peraza-Martinez is responsible for the constant current charging (CCC), Min Hua Wu is responsible for the constant voltage charging (CVC) and Rohan Badiga is responsible for the microcontroller unit (MCU). Overall all of us learned a lot about batteries and different types of battery chargers. Even though we were not able to achieve our original goal of charging two batteries at a time, in the end we were able to charge a single cell nickel-cadmium battery while also following the safe charging rate of C/10, which will fully charge the battery after around 10 hours, without any loss of charge. The CVC was also fun but difficult to learn as I have not taken any power electronics courses and we needed to use a Buck converter in the circuit. Originally, I had the microcontroller unit and Rohan had the constant voltage charging circuit, but during the project we realized that his major is computer engineering so he would be better at the coding part and my major is electrical engineering so I would be better at the hardware part. Due to how impactful Covid-19 has been on every student taking senior design, our group still managed to meet up every week physically in the labs to work on the project. Since not everyone can meet around the same time, we had to do a lot of time management in order to progress with the project.

Rohan Badiga:

The battery charger project was a success. We built a Constant Current Circuit, Constant Voltage Circuit, Discharge Circuit and overall Battery Charger. In the start of this project I was incharge of building the Constant Current Circuit schematic in LTSpice. I ran all the simulation and collected data. Being a Computer Engineering Student, I have limited experience with hardware components. Due to Covid 19 majority of my classes did not have any hardware aspect to them so I wasn't too familiar with it. Therefore, my role in this project evolved to head teh software development. I built all the peripherals for the Arduino and implemented them into the circuits. I did all the testing regarding those components as well. Overall, the opportunity to work on this project did give me a lot of insight on hardware as well. I learned a lot about the different techniques and got hand on experience. My knowledge in embedded systems also expanded as I have never worked with an Arduino before this project. Learning about it was truly an informative experience for me.

12.2 Future Work

Since our original goal was not met, for future expansion we can definitely look into making the battery charger fit two cells of battery, increase the charge rate, and do more safety testing. Making a cheaper design would benefit the impact of this battery charger. The reason cost would be so important other than return on investment is because many people will not see the value of the battery charger if

instead they can simply throw out their old, used batteries and buy new ones. But this adds to e-waste and has harsh impacts for the environment. If the charger is not bought then it doesn't matter how many batteries it could save from landfill; it will never be used. When considering the commercialization of this product it is essential to consider the interface of the charger for the consumer. Another thing that we can possibly implement is that we can use a printed circuit board instead of a breadboard to make sure that the components do not short circuit or fall off while transporting.

12.3 * Acknowledgement

Thanks to professor Roman Chomko for providing guidance and Mr. Manglai Zhou for any technical lab issues that we ran into.

13 * References

Nickel Cadmium Battery Charging

Batteryuniversity.com. 2021. *Charging Nickel-Cadmium Batteries – Battery University*. [online] Available at: <https://batteryuniversity.com/learn/article/charging_nickel_based_batteries> [Accessed 16 March 2021].

Fast and Ultra-fast Chargers

Batteryuniversity.com. 2021. *Fast and Ultra-fast Chargers - Battery University*. [online] Available at: <https://batteryuniversity.com/index.php/learn/article/ultra_fast_chargers> [Accessed 16 March 2021].

Voltage sensor

Youtube.com. 2021. [online] Available at: <https://www.youtube.com/watch?v=dNRzPUBmh_Q> [Accessed 16 March 2021].

Relayt JQC 3FF

Generationrobots.com. 2021. [online] Available at: <https://www.generationrobots.com/media/JQC-3FF-v1.pdf> [Accessed 16 March 2021].

IR2011(gate driver)

Infineon.com. 2021. [online] Available at: <https://www.infineon.com/dgdl/ir2011.pdf?fileId=5546d462533600a4015355c49b831663> [Accessed 16 March 2021].

LM35 (temperature sensor)

Ti.com. 2021. [online] Available at: <https://www.ti.com/lit/ds/symlink/lm35.pdf> [Accessed 16 March 2021].

LM358 (op-amp)

Ti.com. 2021. [online] Available at: <https://www.ti.com/lit/ds/symlink/lm358-n.pdf?ts=1615858416790&ref_url=https%253A%252F%252Fwww.google.com%252F> [Accessed 16 March 2021].

Relay JQC T78

Pdf.voron.ua. 2021. [online] Available at: <https://pdf.voron.ua/files/pdf/relay/JQC-3F(T73).pdf> [Accessed 16 March 2021].

14 * Appendices

Presents information that supplements the design specification, including:

*** Appendix A:** Parts List
*** Appendix B:** Equipment List
*** Appendix C:** Software List (URL to online drive or SVN server, with sharing set to Public. Can omit this appendix if your project didn't involving writing a program)
Appendix D: Special Resources
Appendix E: User's Manual - If your design requires instructions for future use, here is the place to put that information.
Appendix F to Z: Whatever Else You Wish To Add; for instance, here, you may include detailed solution methods or derivations, which you need for your future review of this report or whoever else is interested to pursue this study. Some side drawings and printouts that are of value to people who will continue this work should be given herein. Some information about the vendors and how to locate parts for similar projects must be included herein. In other words, information that is important about the overall construction of the project should be given herein.)

A. Parts List
    a. VCC
        i. IR2011
        ii. Relay JQC-T78
        iii. diode
            1. 1N4001
        iv. mosfet
            1. 2N2222
            2. IRFF510
        v. resistor
            1. 1k
            2. 200k
            3. 100
            4. 470
        vi. inductor
            1. 100u
        vii. capacitor
            1. 100u x2 polarized cap
            2. 470u
            3. 10n
            4. 10u x2 polarized cap
            5. metabolized 1u
            6. ceramic 100n
        viii. op-amp
            1. LM 358
    b. CCC
        i. mosfet

1. 2N3904
2. 2N2222

ii. Resistors

1. 10k
2. 1k
3. 50
4. 100k

iii. op-amp

1. LM 358

c. Discharge

i. Resistor

1. 10

ii. Capacitor

1. 470 uF

d. Relay JQC-3FF
e. LM 35 Temperature Sensor
f. Voltage Sensor Module DC0-25V
g. Arduino Uno

B. Equipment List
a. Mulitimeter
b. Digilink
c. Function Generator
d. DC Power Supply 12V+-
e. DC Power Supply 5V

C. Software
a. Arduino .aio software
b. LT Spice
c. Microsoft Visio

D. Special Resources
a. Mr. Roman Chomko
b. Mr. Manglai

E. User Manual
a. Enter T for Temperature
b. Enter C for Current
c. Enter V for Voltage
d. Enter R for Relay State
e. Enter 0 for Turn off all Relay
f. Enter 1 for Current Circuit
g. Enter 2 for Voltage Circuit
h. Enter 3 for Discharge Circuit

F. Youtube Video Links
a. Constant Current Circuit: https://youtu.be/45ND9ff27s4

b. Constant Voltage Circuit:
https://youtu.be/XED_B1zNt_s
c. Final Demo Video:
https://www.youtube.com/watch?v=IO5YxovEinE